National Security Agency/Central Security Service

Information Assurance Directorate

Application Whitelisting using Microsoft AppLocker®

August 2014

A product of the Mitigations Group

Table of Contents

Figures

Tables

Disclaimer

This Guide is provided "as is." Any express or implied warranties, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the United States Government be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services, loss of use, data or profits, or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this Guide, even if advised of the possibility of such damage.

The User of this Guide agrees to hold harmless and indemnify the United States Government, its agents and employees from every claim or liability (whether in tort or in contract), including attorneys' fees, court costs, and expenses, arising in direct consequence of Recipient's use of the item, including, but not limited to, claims or liabilities made for injury to or death of personnel of User or third parties, damage to or destruction of property of User or third parties, and infringement or other violations of intellectual property or technical data rights.

Nothing in this Guide is intended to constitute an endorsement, explicit or implied, by the U.S. Government of any particular manufacturer's product or service.

Trademark Information

This publication has not been authorized, sponsored or otherwise approved by Microsoft®. Microsoft®, Windows®, AppLocker®, Excel® and Active Directory® are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Introduction

The volume and variety of malware increases on a daily basis. Malware developers and antivirus vendors are in a never-ending arms race. Malware authors continuously modify their creations so they are not detected, and antivirus vendors update their signatures daily to detect new malware variants. Defending against these threats by blocking every known malware sample, a technique known as *blacklisting*, is a reactive technique that does not scale well to the increasing volume and variety of malware. It also does not protect against unknown malware. Many attacks use previously unknown vulnerabilities, also known as zero-day vulnerabilities that cannot be prevented with blacklisting techniques.

Government and corporate networks are prime targets for attackers. They contain valuable proprietary or sensitive information and have a large, diverse attack surface for an adversary to exploit. As operating systems have become more locked down, attacks have shifted from operating systems to applications. This change has left each individual user, and the applications they use, as the main attack vectors into the network.

Application whitelisting is a proactive technique where only a limited number of programs are allowed to run, while all other programs are blocked from running by default. Below are some example scenarios that may be mitigated by using application whitelisting:

- A user receives an enticing email with a link to a program that looks like a greeting card or a streaming video viewer that executes a hidden malicious program.
- A user views a website that silently exploits a previously unknown or unpatched vulnerability in the web browser, or third-party browser add-on, and then downloads a malicious program to further compromise the network and steal data.
- A user opens a document that exploits a vulnerability in the document viewer and an embedded malicious program is extracted and unknowingly executed.
- A user inserts removable media, such as a USB thumb drive, into their computer that automatically executes a malicious program.
- A user installs a program without notifying the administrator, so the program remains unpatched after a critical vulnerability is publicly disclosed and then is exploited by malware.

Since none of the malicious or unauthorized programs in the above scenarios are included in the list of allowed programs, they would not be executed if application whitelisting was enforced. Application whitelisting makes it more difficult for attackers to compromise a network because they must exploit one of the allowed programs on the victim's computer or circumvent the whitelisting mechanism to perform a successful attack. Even if an allowed program is exploited, further malicious activity may still be blocked by the whitelisting mechanism.

Application whitelisting is not a replacement for traditional security software. It should be used as one layer in a defense-in-depth solution. For an application whitelisting solution to be effective:

- All executable code must be blocked by default so only approved programs can run.

- Regular users must not have administrator privileges.
- Users must not be allowed to run programs from directories where they have write access.

Location-based application whitelisting is a form of application whitelisting where large protected locations that contain multiple applications are whitelisted, rather than each application individually. For the best balance of performance, security, and manageability, NSA's Information Assurance Directorate (IAD) recommends employing location-based application whitelisting. The location-based rules restrict which executables and libraries are allowed to execute based on where they reside. The locations must be protected so that only authorized administrators can install or modify the files. This prevents standard users and malicious activities from circumventing the application whitelisting policy. These location-based rules have minimal impact on system performance and allow most program updates and patches to be applied without requiring any rule changes, while preventing the execution of new unauthorized programs and most current malware.

Recent Microsoft Windows operating systems include a feature called AppLocker as an application whitelisting implementation so that administrators can control which files users can run including executables, scripts, Windows Installer files, and Dynamic Link Libraries (DLLs).

About This Guide

This guide describes Microsoft AppLocker settings recommended by the NSA's Information Assurance Directorate (IAD) for deploying location-based application whitelisting on your network. Alternative application whitelisting implementations that may support your organizational needs are commercially available. These alternative implementations may provide support to multiple different operating system platforms for a fee.

This guide also provides administrators with a walkthrough on how to use AppLocker and implement the settings. Using AppLocker for application whitelisting enforcement will not stop all malicious software. It provides an additional layer in a defense-in-depth strategy. The intent of this guidance is to prevent users from unknowingly or accidentally executing malicious code or unauthorized software.

This guide comes with prebuilt rules and scripts. These prebuilt files are designed to make implementation easier and faster and will allow you to make configuration changes that are specific to your environment. Throughout the instructions these prebuilt files will be referenced and used. If for some reason you do not have the files or cannot use the files, instructions on how to configure components to be equivalent to the prebuilt files are included in the appendices. Instructions that reference a prebuilt file will also tell you where to look in the appendices to find instructions that can be used to replace the prebuilt files. If possible it is highly recommended that you use the prebuilt files.

Please read this entire document before implementing the guidance. It is highly recommended to use audit-only mode to deploy the AppLocker policy and understand its impact prior to rule enforcement. Also, it is recommended that any configuration changes and implementation of AppLocker be tested and

validated in a test/lab environment prior to operational or production use to ensure the settings are correct and the system is stable.

All examples in this document use either Windows 7 Ultimate edition or Windows Server 2008 R2 Enterprise edition.

AppLocker Overview

AppLocker is designed to enhance security and not replace traditional security software, such as antivirus and firewalls. AppLocker is available only in certain editions of Windows Server 2012, Windows Server 2008 R2, Windows 8, and Windows 7. AppLocker is not available at all in Windows Server 2003, Windows XP, and other old versions of Windows.

AppLocker enhances the functionality of the older Software Restriction Policies (SRP) feature. SRP was originally designed for Windows XP and Windows Server 2003. AppLocker differs from SRP by utilizing the Windows kernel to enforce the policy, improving the certificate rules for digitally signed software, and creating an audit mode to test the policy prior to enforcement. If SRP and AppLocker are implemented on the same domain, Microsoft recommends that they should be configured in separate Group Policy Objects (GPOs).

AppLocker Limitations

Although AppLocker has some major improvements over SRP, there are some limitations as indicated below.

As seen during some exploitation, application whitelisting does not provide protection when injecting controlled files directly into memory since the files are opened for non-executable data access, but then used for execution instead[1].

According to Microsoft, AppLocker is not suited for business groups or organizations that have to install applications on a case-by-case basis without an approval from the Information Technology (IT) department. However, this process inherently contradicts all forms of centralized software control and is not recommended.

Why Use AppLocker?

In today's society, some of the most successful malware exploitation involves Trojan horses, where the user installs the program under false pretenses. Application whitelisting implementations, such as AppLocker, can help mitigate these types of attacks by restricting the files that users or groups are allowed to run. AppLocker allows IT administrators to create rules to allow or deny programs based

[1] *National Cyber Security Centre: Application Whitelisting with Microsoft AppLocker*. Jun 2012: 4. New Zealand National Cyber Security Centre. Web. 21 Aug 2012.

upon trusted users, trusted paths, trusted file hashes, and/or trusted publishers. Access control technologies like access control lists (ACLs) help control the users' access to files and other system resources. However, when a user runs a process, that process uses the same level of access to data as the user. Therefore, information/files can easily be modified, deleted, or transmitted due to a user knowingly or unknowingly running malicious software. AppLocker assists in mitigating these types of attacks by restricting the files that users or groups can execute.

Some software publishers create "portable" applications that standard users (non-administrators) can run without installing. This type of software deployment can violate an organization's written security policy and circumvent traditional application deployment solutions that allow software to be installed only in controlled locations. AppLocker helps administrators prevent per-user applications from running by allowing administrators to create rules that deny such files from executing.

Use the list below to determine if your organization is suitable to use AppLocker as part of your organization's application control solution:

- Deployed or plan to deploy Windows Server 2012, Windows Server 2008 R2, Windows 8 (only Enterprise edition), or Windows 7 (only Ultimate or Enterprise editions) across your whole organization. If you will continue to have many older Windows versions (e.g., Windows XP) as well, consider using SRP instead, since it can enforce a single policy for both the older and newer Windows versions.
- A policy stating that only authorized applications may be run.
- A need to improve control over your organization's applications to enforce your authorized application policy.
- A standard application baseline and a robust application installation, management and issue resolution process.
- The ability to test policies against the organization's requirements.

Advantages

Some of the advantages of Microsoft's AppLocker include:

- Included at no additional cost within Windows Server 2012, Windows Server 2008 R2, Windows 8 (only Enterprise edition) and Windows 7 (only Ultimate and Enterprise editions)
- Manageable through Group Policy Objects
- Easily importable and exportable policies
- Prevents use of unauthorized applications
- Does not require daily definition updates
- May reduce administrative overhead by reducing the number of conflicting or misconfigured applications.

Disadvantages

Some of the disadvantages of Microsoft's AppLocker include:

- Reporting is limited to local event log messages, unless Windows Event Forwarding is enabled or a Security Information and Event Management (SIEM) solution is used
- Requires some performance overhead to enforce the whitelist (varies depending on rule type)
- Only available for certain Windows versions and editions

Deploying AppLocker

AppLocker supports four types of rules: Path Rules, File Hash Rules, Publisher Rules, Packaged App Rules. The rule types can be applied to restrict executables (.exe), scripts (.ps1, .bat, .cmd, .vbs, .js), installer files (.msi), libraries (.dll), and packaged apps (.appx). The Packaged App Rules are only supported on Windows 8 and Windows Server 2012 and new versions. Each file type has a separate rule collection within AppLocker and can be configured individually.

Path Rules allow the restriction of program execution based on where the program is located on disk or over the network. Administrators can create rules for a specific file or an entire directory tree. One potential difficulty with this rule type is that users may have authorized applications installed in diverse locations and either need multiple rules to allow all the possible locations or have administrators migrate them into common locations. Depending on the diversity of the environment, it can be time-consuming to keep track of or migrate all of the legitimate program folders. In addition, since the rules are only based on the location of the files and not on their content, the access controls applied to those locations need special attention to ensure that regular users are restricted from adding or modifying the files in those locations which are then allowed to execute.

File Hash Rules use a cryptographic hash to uniquely identify a specific executable file. This rule is most useful for unsigned files that are not in whitelisted locations identified by a path rule that are needed for an application to function properly. Please note for this rule type, the rule must be modified whenever the program is updated because the hash changes with any change to the file's contents. Therefore, due to the ongoing management burden of keeping the hash rules updated, this rule is only recommended for use in certain situations when the other rule types cannot be used.

Publisher Rules allow for greater flexibility, but can only be used to identify software files that are digitally signed. Most recent applications have digital signatures that can be used for Publisher Rules, but unfortunately many others still do not. In order for a Publisher Rule to effectively allow execution of a program, all of the program's executable files must be digitally signed for them all to be allowed to run and function properly. Publisher Rules can identify all files signed by a publisher, or they can be further restricted to the product name, file name, or even to a minimum file version. Note that the file version is listed separately in the properties for each file and may not be the same as the program version or consistent across all files of a particular product. Users cannot easily circumvent this rule by just renaming the file since AppLocker gets this information from the digital signature of the executable. The primary concern with this rule type is that there are many applications that are not digitally signed.

Packaged App Rules allows a single AppLocker rule to control whether the entire app, also known as a Windows app, is allowed to run. This rule permits all files within an app package to share the same identity and be controlled by AppLocker together. Windows AppLocker only supports Packaged Apps that are signed by the software publisher. Since all Packaged Apps can be installed by a standard user, it is recommended to create specific policies to restrict Packaged Apps to only the authorized ones. Appendix G – Packaged App Rule is a brief instruction on creating a whitelist for Packaged Apps. For additional information about Packaged Apps see Microsoft's Packaged Apps and Packaged App Installer Rules in AppLocker.

Configuring AppLocker

There are several different ways to configure AppLocker depending on whether you want to use local or group policies. You can use the Group Policy Management Console (GPMC), the Local Group Policy Editor, or the Local Security Policy Editor. The steps below will walk you through how to find the AppLocker configuration section in each tool.

Group Policy Management Console

The Group Policy Management Console (GPMC) is included with Windows Server® 2008 and later. This feature is not installed by default with the operating system. You can use the Server Manager to install GPMC.[2]

1. Click on **Start**, type ***gpmc.msc*** in the search field and select **ENTER**.

   Or

   Click **Start**, go to **Administrative Tools** and click **Group Policy Management** to open the GPMC.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. Select the desired GPO. Expand the console to **Computer Configuration > Policies > Windows Settings > Security Settings > Application Control Policies** > **AppLocker**.

Local Group Policy Editor

1. Click **Start**, type ***gpedit.msc*** in the search field and select **ENTER**.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. In the console, go to **Computer Configuration > Windows Settings > Security Settings > Application Control Policies > AppLocker**.

Local Security Policy Editor

1. Click **Start**, type ***secpol.msc*** in the search field and select **ENTER**.

   Or

   Click **Start** go to **Control Panel**. Click **System and Security**, then click **Administrative Tools** and double-click **Local Security Policy**.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. In the console, expand to **Security Settings > Application Control Policies** > **AppLocker**.

[2] Microsoft Windows Server: Install the GPMC. 20 Feb 2013. http://technet.microsoft.com/en-us/library/cc725932.aspx.

Implementation

Prior to configuring AppLocker policies, there should be a design and planning process to understand the potential effects the AppLocker rules will have on your system. The next few steps will enable you to configure enforcement for AppLocker rules. The three enforcement modes include the following: (1) not configured, (2) enforce rules, and (3) audit only. The not configured mode is Microsoft's default setting where AppLocker is not configued and is not enforced. The enforce rules mode is when the rules are applied to enforce the application whitelisting policy defined by the rule collection. Audit only mode does not enforce the rules, but simulates applying the rules. It logs which files would be blocked by the proposed policy. The following sections explain the procedures for how to configure the recommended AppLocker policy and tailor it to your system.

AppLocker can be cirumvented in several ways. It is recommended to limit or remove administrator privileges from users so unauthorized administrators do not have the ability to make unplanned changes to the AppLocker policies or disable the required AppID Service. Another way to circumvent the intended application whitelisting policy is to add or modify the files that AppLocker allows to be executed. Therefore, it is essential that all executable files and containing directories identified by the AppLocker rules be protected from user modification by file system access controls or other means (e.g., Host Intrustion Prevention System (HIPS) rules).[3]

Starting the Application Identity Service

Prior to using AppLocker, the Application Identity service must be started on each computer in order to enforce an AppLocker policy. AppLocker employs this user-mode service to aid the kernel-mode AppLocker implementation. This service can be configured to start automatically by an administrator using Group Policy settings. This can be applied either in local Group Policy for a single computer using the Group Policy Editor (*gpedit.*msc) or by domain-based Group Policy via the Group Policy Management Console (*gpmc.msc*) and editing a policy. *(GPMC is only available on Windows client computers by installing the Remote Server Administration Tools and on Windows servers by installing the Group Policy Management feature.)*

1. Open the applicable Group Policy in a Group Policy Editor tool for editing.
2. Browse to **Computer Configuration > Policies > Windows Settings > Security Settings > System Services**.
3. Double-click the **Application Identity** service to open its **Properties** window.
4. In the **Application Identity Properties** window, check **Define this policy setting**, select **Automatic** in the Select service startup mode list, and then click **OK**.

[3] See the section in "Implementing DSD's Top Four For Windows Environments" on "Using accesschk to Test Permissions" for checking that standard users do not have write access to whitelisted locations: http://www.dsd.gov.au/publications/Implementing_Top_4_for_Windows.pdf.

Enabling DLL Protection

Even though the DLL rule collection is not enabled by default, it is recommended that the AppLocker application whitelisting policy be configured for libraries as well, since they are used to run code and can be used to easily bypass the intended application whitelisting policies. Importing the starter policy file will automatically enable the DLL rule collection and configure it for auditing. Refer to Appendix B – Enabling the DLL Rule Collection for instructions for manually enabling the DLL rule collection.

Auditing

Auditing is the first major step that you will take to implement application whitelisting. This will allow you to generate logs that will be useful in tailoring the policy for your organization. In the auditing phase you will create an application whitelisting test group, create a test policy, import the policy, verify that logging is enabled, apply the policy, and update the test clients with the new test policy.

Prior to enforcing AppLocker Rules, auditing should be used to verify that the AppLocker policy will function as expected and not adversely affect authorized applications. This is useful for testing new rules prior to rule deployment, as well as, helping to determine whether any additional rules are necessary for your policies. Once it is determined that the rules are behaving correctly, then the policy should be changed to enforce the rules. Each rule collection (executable, installer, script, and DLL) can be configured separately for auditing or enforcement.

The auditing step should allow you to determine which applications are running on your network. It will not block any activity in your environment, as auditing mode will only monitor system activity, simulate the proposed policy, and log the events which would be blocked if the policy was enforced. If the domain or organizational unit (OU) for AppLocker is considered large then you should audit a small group of computers before auditing the entire domain or OU. Auditing a small group of computers first should keep the event collection server from being overwhelmed by a large amount of logs. The auditing group needs to be as diverse as possible so that it will contain a representation of the different configurations throughout your domain or OU. Later, once the policy has been tailored to reduce false positive events, you can audit your entire domain or OU.

Create an AppLocker Group Policy

To begin auditing create a new group policy for AppLocker settings.

1. Open the **Group Policy Editor**.
2. From the taskbar, select ***Action > New***.
3. Give your new group policy a descriptive name based on your network's naming conventions and click **OK**.

Create an Audit Group

If you are going to audit a small group before auditing your entire domain or OU and need to create a new group, then follow the steps below in the sections: Create a New Audit Group, Add Computers to the New Audit Group, and Link New Audit Group to Group Policy. These sections can be skipped if you are going to audit an existing group or your entire domain or OU.

Create a New Audit Group

1. Open the **Server Manager**.
2. Browse to ***Roles > Active Directory Domain Services > Active Directory Users and Computers > <YOUR DOMAIN NAME>***.
3. Right click on the Organizational Unit (folder) or domain that your group will go in and then choose ***New*** > ***Group***.
4. Type in a group name and then press **OK**.

Note: You can use an existing group if it accurately represents the different configurations across the domain or OU and the existing group is small enough that it will not create too many log entries during initial auditing.

Add Computers to the New Audit Group

Add all computers that you have selected to be part of the newly created audit group.

1. Open the **Server Manager**.
2. Browse to ***Roles > Active Directory Domain Services > Active Directory Users and Computers > <YOUR DOMAIN NAME>***.
3. Open the organizational unit (folder) that contains the computer you want to add.
4. Right click on the computer(s) and select **Add to Group**.
5. Type in the name of your AppLocker audit group and then press **OK**.
6. Repeat until all computers have been added.

Link New Audit Group to Group Policy

1. Open the **Group Policy Editor**.
2. In the group policy editor browse to your domain and then select the folder **Group Policy Objects**.
3. Select the AppLocker group policy that you created.
4. In the details pane select the **Scope** tab.
5. Press **Add** under **Security Filtering.**
6. Type in the name of the AppLocker auditing group that you created then press **OK**.

Configuring the Starter Policy for Auditing

Included with this guide is an XML file called “Starter AppLocker Policy.xml” that can be imported to set up the initial recommended location-based AppLocker policy. Configure the recommended location-based AppLocker starting rules to create an application whitelisting policy for initial auditing. These rules are intended to allow users to run applications that have been properly installed by an administrator, while denying users from accidentally or unknowingly running unauthorized or malicious programs. By analyzing the audit events and tailoring the AppLocker policy prior to enforcement, you can be assured that commonly used authorized programs will continue to function properly after switching to enforcement.

Note that this starter policy also includes rules for the best practice of restricting administrator accounts from browsing the web or reading email by denying them from easily running web browsers and email

clients (see Appendix C – AppLocker Rules to Prevent Administrators from Easily Browsing the Internet/Email for more information).

To import the AppLocker policy XML file:

1. In the AppLocker console tree, right-click **AppLocker**, and then click **Import Policy**….
2. Browse to the location where the AppLocker policy XML file is saved, select it and click **Open**.

This loads the starter policy with the following location-based application whitelisting rules:

Executable Rules

Rule Name	Action	User or Group	Condition Type	Condition / Exception Value
Allow everyone to execute all files located in the Program Files folder	Allow	Everyone	Path	%PROGRAMFILES%*
Allow everyone to execute all files located in the Windows folder	Allow	Everyone	Path	%WINDIR%*
exception to the preceding rule			Path Exception	%SYSTEM32%\catroot2*
exception to the preceding rule			Path Exception	%SYSTEM32%\com\dmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\FxsTmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\drivers\color*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\PRINTERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\SERVERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Debug*
exception to the preceding rule			Path Exception	%WINDIR%\PCHEALTH\ERRORREP*
exception to the preceding rule			Path Exception	%WINDIR%\Registration*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\com\dmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\FxsTmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Temp*
exception to the preceding rule			Path Exception	%WINDIR%\tracing*
Allow administrators to execute all files	Allow	BUILTIN\Administrators	Path	*

Table 1: Starter Executable Rules

Installer Rules

Rule Name	Action	User or Group	Condition Type	Condition / Exception Value
Allow everyone to run all Windows Installer files located in the Windows\Installer folder	Allow	Everyone	Path	%WINDIR%\Installer*
Allow administrators to run all Windows Installer files	Allow	BUILTIN\Administrators	Path	*

Table 2: Starter Installer Rules

Script Rules

Rule Name	Action	User or Group	Condition Type	Condition / Exception Value
Allow everyone to run all scripts located in the Program Files folder	Allow	Everyone	Path	%PROGRAMFILES%*
Allow everyone to run all scripts located in the Windows folder	Allow	Everyone	Path	%WINDIR%*
exception to the preceding rule			Path Exception	%SYSTEM32%\catroot2*
exception to the preceding rule			Path Exception	%SYSTEM32%\com\dmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\FxsTmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\drivers\color*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\PRINTERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\SERVERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Debug*
exception to the preceding rule			Path Exception	%WINDIR%\PCHEALTH\ERRORREP*
exception to the preceding rule			Path Exception	%WINDIR%\Registration*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\com\dmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\FxsTmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Temp*
exception to the preceding rule			Path Exception	%WINDIR%\tracing*
Allow administrators to run all scripts	Allow	BUILTIN\Administrators	Path	*

Table 3: Starter Script Rules

DLL Rules

Rule Name	Action	User or Group	Condition Type	Condition / Exception Value
Allow administrators to execute all DLLs	Allow	BUILTIN\Administrators	Path	*
Allow everyone to execute all DLLs located in the Program Files folder	Allow	Everyone	Path	%PROGRAMFILES%*
Allow everyone to execute all DLLs located in the Windows folder	Allow	Everyone	Path	%WINDIR%*
exception to the preceding rule			Path Exception	%SYSTEM32%\catroot2*
exception to the preceding rule			Path Exception	%SYSTEM32%\com\dmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\FxsTmp*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\drivers\color*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\PRINTERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\spool\SERVERS*
exception to the preceding rule			Path Exception	%SYSTEM32%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Debug*
exception to the preceding rule			Path Exception	%WINDIR%\PCHEALTH\ERRORREP*
exception to the preceding rule			Path Exception	%WINDIR%\Registration*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\com\dmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\FxsTmp*
exception to the preceding rule			Path Exception	%WINDIR%\SysWOW64\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Tasks*
exception to the preceding rule			Path Exception	%WINDIR%\Temp*
exception to the preceding rule			Path Exception	%WINDIR%\tracing*

Table 4: Starter DLL Rules

Notice that there are numerous exceptions listed as part of the starter rules to deny users the ability to execute files from user writable subdirectories of whitelisted directories. For example, "%WINDIR%*" is whitelisted to allow users to execute files that reside there, but "%WINDIR%\Temp*" is listed as an exception in order to prevent users from executing the files that reside there because the default file system access controls allow any authenticated user to add new files to that subdirectory.

If you do not want to import the starter rules, then you can use the recommended location-based AppLocker starting rules found in Appendix A to create an application whitelisting policy for initial auditing.

Reviewing Logs

Configure the auditing group to forward AppLocker log events to a collection server. These logs can be viewed together on the collection server. Allow several days for logs to be generated so that they can reflect everyday use of the systems.

It is recommended that your event collection server collect the following AppLocker events: 8003, 8004, 8006, and 8007. Table 5 has additional information from Microsoft on possible AppLocker events and their meanings. For additional details on forwarding Windows events and reviewing log entries, refer to Microsoft's documentation and see the guide: "*Spotting the Adversary with Windows Event Log Monitoring*" on the NSA website.

Event ID	Level	Event Message	Description
8000	Error	Application Identity Policy conversion failed. Status *<%1>*	Indicates that the policy was not applied correctly to the computer. The status message is provided for troubleshooting purposes.
8001	Information	The AppLocker policy was applied successfully to this computer.	Indicates that the AppLocker policy was successfully applied to the computer.
8002	Information	*<File name>* was allowed to run.	Specifies that the .exe or .dll file is allowed by an AppLocker rule.
8003	Warning	*<File name>* was allowed to run but would have been prevented from running if the AppLocker policy were enforced.	Applied only when the **Audit only** enforcement mode is enabled. Specifies that the .exe or .dll file would be blocked if the **Enforce rules** enforcement mode were enabled.
8004	Error	*<File name>* was not allowed to run.	Access to *<file name>* is restricted by the administrator. Applied only when the **Enforce rules** enforcement mode is set either directly or indirectly through Group Policy inheritance. The .exe or .dll file cannot run.
8005	Information	*<File name>* was allowed to run.	Specifies that the script or .msi file is allowed by an AppLocker rule.
8006	Warning	*<File name>* was allowed to run but would have been prevented from running if the AppLocker policy were enforced.	Applied only when the **Audit only** enforcement mode is enabled. Specifies that the script or .msi file would be blocked if the **Enforce rules** enforcement mode were enabled.
8007	Error	*<File name>* was not allowed to run.	Access to *<file name>* is restricted by the administrator. Applied only when the **Enforce rules** enforcement mode is set either directly or indirectly through Group Policy inheritance. The script or .msi file cannot run.

Table 5: Microsoft's Table of Events Affected by AppLocker Rules[4]

[4] Event table from Microsoft website, 21 Dec 2012. http://technet.microsoft.com/en-us/library/ee844150%28v=ws.10%29.aspx

The AppLocker event data include several fields that are useful to review during auditing and provide essential information for tailoring the policy. In the event details, the PolicyName field indicates which Rule Collection triggered the rule (i.e., EXE for Executable Rules, DLL for DLL Rules, MSI for Windows Installer Rules, or Script for Script Rules). If the Fqbn field has data, then the file is digitally signed and could be allowed to run using a Publisher rule. The FilePath field contains the full path to the affected file, often containing the AppLocker specific variables listed in the table below as part of the path.

Windows directory or drive	AppLocker path variable	Windows environment variable
Windows	%WINDIR%	%SystemRoot%
System32	%SYSTEM32%	%SystemDirectory%
Windows installation directory	%OSDRIVE%	%SystemDrive%
Program Files	%PROGRAMFILES%	%ProgramFiles% and %ProgramFiles(x86)%
Removable media (for example, CD or DVD)	%REMOVABLE%	
Removable storage device (for example, USB flash drive)	%HOT%	

Table 6: Microsoft's Table of Events Affected by AppLocker Rules[5]

The TargetUser and TargetProcessId fields in the event details may also be useful, but they are not very helpful by themselves. See Appendix E for more information about using a script to fill in more useful event details based on these fields.

Using Event Viewer with AppLocker

The Event Viewer is available for reviewing the logs for AppLocker events, which contain information about the applications that are affected by AppLocker rules. AppLocker logs both allowed and denied file execution, so the default AppLocker event log is mostly full of allowed events. Creating a custom view in Event Viewer will enable automatic filtering to view only the warning and error events that are generated. To create this custom view:

1. Click **Start**, type ***Event Viewer*** (or ***eventvwr.msc***), and then press ENTER; or use the following:
    a. Click **Start**, and then click **Control Panel**.
    b. Click **System and Security**, and then click **Administrative Tools**.
    c. Double-click **Event Viewer**.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. In the Event Viewer window, click **Action > Import Custom View**….
4. Select the saved **AppLocker Errors Event Viewer Custom View.xml** that is included with this guide and click **Open**.
5. Browse to **Custom Views > AppLocker Warnings and Errors**.

[5] AppLocker variables table from Microsoft website, 21 Dec 2012. http://technet.microsoft.com/en-us/library/dd759068.aspx

Once AppLocker events have triggered, a list of the AppLocker events should be displayed. The events include information about the user, computer, file name, and date and time. Review the event logs to determine how to tailor the policy as discussed in the following section.

Tailoring Rules

This section will guide you in determining if you are required to create or modify AppLocker rules based on the logged events. The exceptions or rules you create will be based on if an authorized application needs a new rule or exception in order to run properly. Every time you tailor the policy make sure to save, label, and test it. Ensure you document all your findings to diligently manage your AppLocker policy.

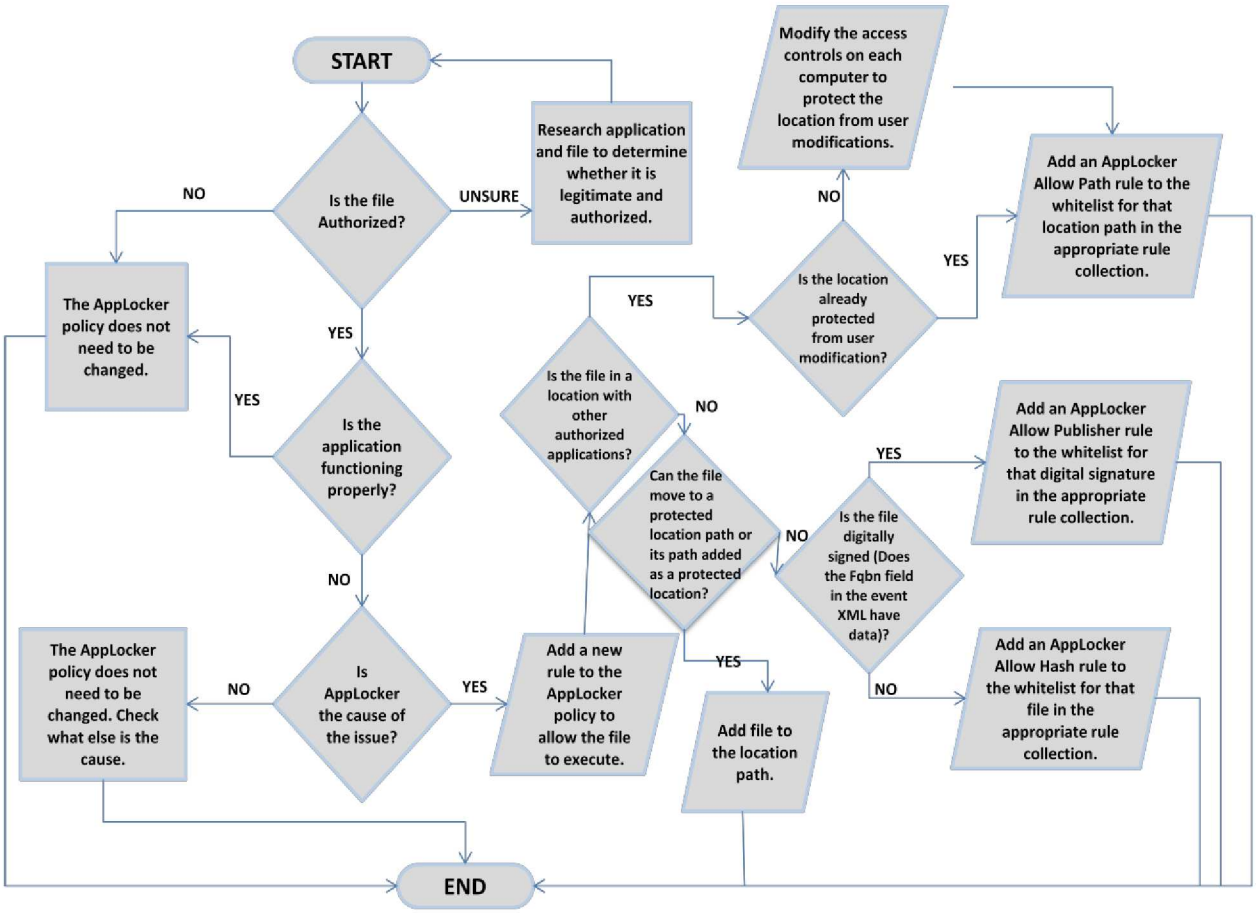
YOU SHOULD WAIT UNTIL YOU HAVE A SUFFICIENT AMOUNT OF LOGS GENERATED BY THE TEST POLICY BEFORE YOU ATTEMPT THIS STEP. IN ORDER TO CREATE EFFECTIVE RULES AND/OR EXCEPTIONS, YOUR LOGS NEED TO HAVE ENOUGH DATA TO REFLECT NORMAL SYSTEM USAGE.

FOR EASE OF IMPLEMENTATION DURING INITIAL TAILORING ASSUME THAT LOGGED APPLICATION BEHAVIORS ARE LEGITIMATE. LATER DURING ENFORCEMENT ASSUME THE OPPOSITE.

Decision Tree

The following decision tree will help you decide if you need to create or modify AppLocker rules to allow an application to run. Links are included in the process to specific instructions in this document that show how to create or edit specific rules.

IT IS IMPORTANT WHEN CREATING RULES TO MAKE THEM NEITHER TOO GENERAL NOR TOO SPECIFIC. IF RULES ALLOW MORE FLEXIBILITY THAN IS NEEDED, IT COULD LEAD TO APPLICATION WHITELISTING BEING INEFFECTIVE AND EASILY CIRCUMVENTED. IF RULES ARE TOO SPECIFIC, THEN THE POLICY BECOMES BRITTLE AND REQUIRES BURDENSOME MANAGEMENT OVERHEAD AND MAINTENANCE, AS MANY FILES MAY NEED TO BE INDIVIDUALLY LISTED AS PART OF AN EXCEPTION AND IF THERE ARE ANY SLIGHT CHANGES TO THE APPLICATION, THE RULES MAY NO LONGER PROPERLY APPLY.



Visual Decision Tree

Text Version of the Decision Tree

Start with answering the questions to determine what to do with an application that may need an exception.

1. Is the file authorized?
   1.1. If you are unsure, research the file to determine if it is legitimate for your organization or system. Once you have your answer, start over from the beginning.
   1.2. If the file is not authorized for execution, then AppLocker should block it from executing and no change to the policy is needed.
   1.3. If the file is legitimate and authorized, does the application function properly?
      1.3.1. If the answer is yes, then no additional rules or exceptions are required since the application is functioning properly.
      1.3.2. If the answer is no, is AppLocker the cause of the problem? Check the logs and use the Event IDs to help determine if AppLocker is the cause (see Reviewing Logs).

         Note: Applications that should be allowed to run can be blocked for the following reasons:

         - A deny rule is preventing the file from running.
         - There is no rule to allow it to run.

- An existing rule is too restrictive which prevents the file from running.

1.3.2.1. If the answer is no, then check what else is preventing the application from functioning properly and resolve that issue.

1.3.2.2. If the answer is yes, then is the file in a location with other authorized applications and files that will need to execute as well?

1.3.2.2.1. If the answer is yes, then is the location already protected from a user adding or modifying the files there?

1.3.2.2.1.1. If the answer is yes, then that location can be added to the whitelist by adding a new AppLocker Allow Path rule in the appropriate rule collection as indicated by the PolicyName field in the event for that general location.

1.3.2.2.1.2. If the answer is no, then change the access permissions on the location to disallow users from adding or modifying files in that location. Once the location is adequately protected, it can be added to the whitelist by adding a new AppLocker Allow Path rule in the appropriate rule collection as indicated by the PolicyName field in the event for that general location.

1.3.2.2.2. If the answer is no, can the file(s) move to an authorized path location or can its current path be added as an authorized path location?

1.3.2.2.2.1. If the answer is yes, move the file(s) to an authorized path location or add an AppLocker rule to allow the path location in the whitelist, as well as, ensure it is protected from user modification.

1.3.2.2.2.2. If the answer is no, then is the file digitally signed and is it likely to be patched and/or have updates in the future?

Note: You can tell whether the file was digitally signed by checking the event details to see if there is data in the Fqbn field. If the Fqbn field has data, then the file that triggered the event was digitally signed.

1.3.2.2.2.2.1. If the answer is yes, then add a new AppLocker Allow Publisher rule in the appropriate rule collection as indicated by the PolicyName field in the event. If you have access to the file, you can browse to the file so that AppLocker can automatically fill in the Publisher properties with the appropriate information. If you do not have access to the file, then you may need to manually export and edit the XML for the AppLocker policy to set the PublisherName attribute value on the FilePublisherCondition element to the Fqbn data from the event details.

1.3.2.2.2.2.2. If the answer is no, then add a new AppLocker Allow File hash rule in the appropriate rule collection as indicated by the PolicyName field in the event. If you have access to the file, you can browse to the file so that AppLocker can automatically fill in the file hash information. If you do not have access to the file, then you

may need to manually export and edit the XML for the AppLocker policy to set the Data attribute value on the FileHash element to the FileHash data from the event details.

2. Click on a link below to go to instructions in this document on how to create or modify AppLocker rules. The instructions are located in

3. Appendix D – Creating and Modifying AppLocker Rules of this document.

Create AppLocker Rules

Edit AppLocker Rules

Delete AppLocker Rules

Audit Entire Domain or OU

If you have created a test group for auditing, you should now expand your auditing group in phases until you turn on auditing for your entire domain or OU. You must audit the entire domain or OU to be sure that you did not miss any applications that might not have been used within the audit group.

To fully deploy AppLocker Rules:

1. Open the **Group Policy Management** console.
2. Select the AppLocker group policy that you created.
3. Under **Security Filtering**, remove the AppLocker audit group that you created.

After the AppLocker rules have been deployed more broadly, repeat the Reviewing Logs and Tailoring Rules steps for your entire domain or OU.

Create AppLocker Alert on User Computers

AppLocker does not always display a pop-up alert when a file is blocked from executing, especially for libraries. The event will be logged, but it will be difficult for users to understand why their application may not be functioning properly. Therefore, it would be helpful for users to be alerted to when AppLocker blocks any file. An administrator can import a scheduled task to trigger instant pop-up alerts for AppLocker by following the steps below.

1. Click **Start**, type ***Task Scheduler*** *(or* **taskschd.**msc) as an administrator, and then press **ENTER**; or use the following:
    a. Click **Start** and then click **Control Panel**.
    b. Click **System and Security** and then click **Administrative Tools**.
    c. Double-click **Task Scheduler**.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. Click **Action > Import Task.**
4. Select the "AppLocker Popup Alert.xml" file that is included with this guide and click **Open**.

In order to push this task out to all the Windows 7 computers, you may want to push out a script via Group Policy to each computer to install the task XML file locally. A sample batch script is included with this guide, called "Create AppLocker Popup Task.bat" that can be set as a Computer Startup script with the task XML file located in the domain's SYSVOL\<Domain Name>\Scripts file share. This location may be changed within the batch script, as long as every computer account has access to the location in order to load the task XML file.

Enforce AppLocker Rules

Once sufficient auditing and tailoring have been performed so that there are no more false positive AppLocker events for authorized files, it is time to switch to enforcing the AppLocker policy. As before, start with a small test group to begin the enforcement to be sure that there are no unforeseen issues and then expand the group until the entire domain or OU is enforcing an appropriate application whitelisting policy using AppLocker. Follow the steps below to change the policy from audit mode to enforcement mode.

1. Within the AppLocker Group Policy Editor, right-click **AppLocker** and then click **Properties**.

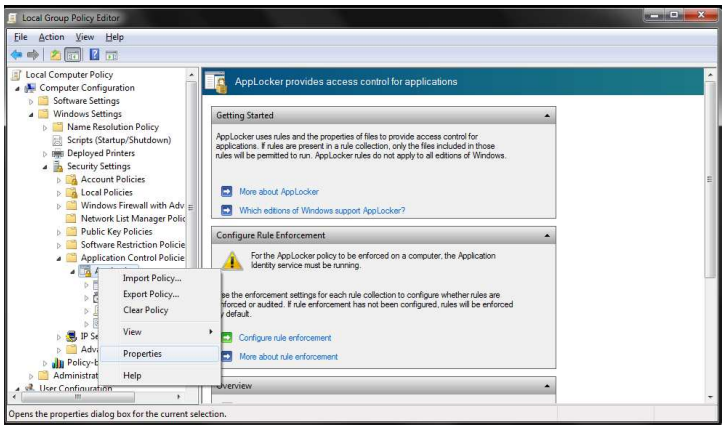


Figure 1: Enforce AppLocker Rule

2. On the **Enforcement** tab, select **Enforce rules** in the drop-down list for a rule collection.

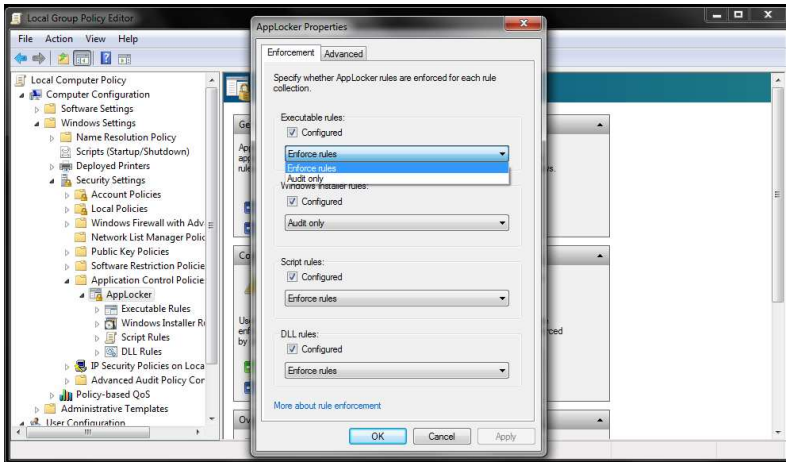


Figure 2: Enforcement Tab

3. Repeat step 2 to configure enforcement for each rule collection.
4. Click **OK** or press **ENTER**.

Where to Find More Information?

Here are some other guides about application whitelisting using AppLocker.

AppLocker Deployment by Microsoft:
http://www.microsoft.com/en-us/download/details.aspx?id=28372

AppLocker Step-by-Step Guide by Microsoft:
http://technet.microsoft.com/en-us/library/dd723686(v=ws.10).aspx

AppLocker Operations Guide by Microsoft:
http://technet.microsoft.com/en-us/library/ee791916(v=ws.10).aspx

Microsoft's Packaged Apps and Packaged App Installer Rules in AppLocker:
http://technet.microsoft.com/en-us/library/hh831350.aspx

Application Whitelisting Explained by Australia's Cyber Security Operations Centre:
http://www.dsd.gov.au/publications/csocprotect/Application_Whitelisting.pdf

Implementing DSD's Top Four for Windows Environments - APPENDIX B: APPLOCKER IMPLEMENTATION NOTES by Australia's Defence Signals Directorate:
http://www.dsd.gov.au/publications/Implementing_Top_4_for_Windows.pdf

Application Whitelisting with Microsoft AppLocker by New Zealand's National Cyber Security Centre:
http://ncsc.govt.nz/sites/default/files/articles/NCSC%20Applocker-public%20v1.0.5.pdf

Appendix A – Configuring the Starter AppLocker Policy

Follow the steps below to configure the starter AppLocker policy for initial auditing:

1. If you're already in the Group Policy Editor and AppLocker skip to Step 3, or else click **Start**, type ***gpedit.msc***, and then press ENTER.
2. Click **Yes**, if the **User Account Control** dialog box appears to confirm the action it displays.
3. In the console, go to **Computer Configuration** > **Windows Settings** > **Security Settings** > **Application Control Policies**.

Generic AppLocker Settings
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Advanced tab → check the "Enable the DLL rule collection" checkbox
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Executable rules group, check the "Configured" checkbox
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Executable rules group, select "Audit only" from the drop-down list
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Windows Installer rules group, check the "Configured" checkbox
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Windows Installer rules group, select "Audit only" from the drop-down list
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Script rules group, check the "Configured" checkbox
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the Script rules group, select "Audit only" from the drop-down list
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the DLL rules group, check the "Configured" checkbox
Security Settings\Application Control Policies\AppLocker → Configure Rule Enforcement → Enforcement tab → in the DLL rules group, select "Audit only" from the drop-down list

Table 7: Starter AppLocker Settings

In the table above are the starter AppLocker rules. This is not a comprehensive list of rules, and the rules that will be needed will vary based on your organizational needs. Remember, you can automatically import this starter policy by importing the XML file included with this guide instead of manually creating the rules below.

1. Within the AppLocker Group Policy Editor, highlight **Executable Rules** and then right-click on **Executable Rules**.

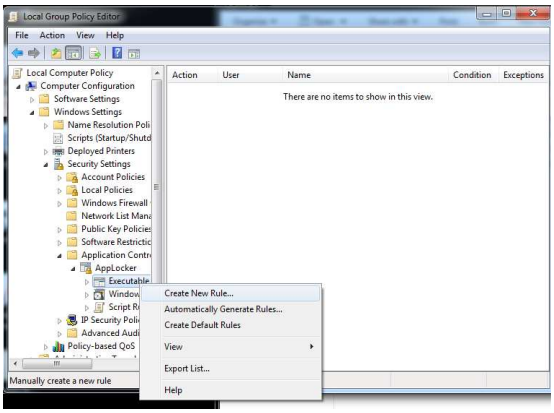


Figure 3: Create New Rule

2. Select **Create New Rules**.
3. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Everyone,*** press **ENTER** and then select **Next**.
4. For Conditions, select **Path**, and then select **Next**.
5. Next type in ***%PROGRAMFILES%**** and then select **Next**.
6. There are no exceptions for this rule therefore, you can click **Next** again.
7. Under name, make one up or just use "***Allow everyone to execute all files located in the Program Files folder***" and then click **Create**.
8. If there's a permission window pop-up, select **Yes**.
9. For the next rule, right-click **Executable Rules** and select **Create New Rules**.
10. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Everyone,*** press **ENTER** and then select **Next**.
11. For Conditions, select **Path**, and then select **Next**.
12. Next type in ***%WINDIR%**** and then select **Next**.
13. Under **Exceptions**, the following will be exceptions for the path condition.
    a. The first exception type ***%SYSTEM32%\catroot2**** and either press **ENTER** or click **OK**.
    b. Under Add exception select path, then click **Add**
    c. Now add each of the following path exceptions below:
        1. %SYSTEM32%\com\dmp*
        2. %SYSTEM32%\FxsTmp*
        3. %SYSTEM32%\spool\drivers\color*
        4. %SYSTEM32%\spool\PRINTERS*
        5. %SYSTEM32%\spool\SERVERS*
        6. %SYSTEM32%\Tasks*
        7. %WINDIR%\Debug*
        8. %WINDIR%\PCHEALTH\ERRORREP*
        9. %WINDIR%\Registration*

10. %WINDIR%\SysWOW64\com\dmp*
11. %WINDIR%\SysWOW64\FxsTmp*
12. %WINDIR%\SysWOW64\Tasks*
13. %WINDIR%\Tasks*
14. %WINDIR%\Temp*
15. %WINDIR%\tracing*

14. After you enter the last exception, select **Next**.
15. Under name, make one up or just use "***Allow everyone to execute all files located in the Windows folder***" and then click **Create**.
16. If there's a permission window pop-up, then select **Yes**.
17. For the next rule, right-click **Executable Rules** and select **Create New Rules**.
18. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Administrator***, press **ENTER,** and then select **Next**.
19. For Conditions, select **Path**, and then select **Next**.
20. Next type in "*****" then select **Next**. There are no exceptions for this rule; therefore, you can click **Next** again.
21. Under name, make one up or just use "***Allow administrators to execute all files***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
22. For the next rule, right-click **Windows Installer Rules** and select **Create New Rules**.
23. Under **Permission**, select **Allow**. For **User or Group**, if ***Everyone*** is not the default, click **Select**, type ***Everyone***, press **ENTER**, and then select **Next.**
24. For Conditions, select **Path**, and then select **Next**.
25. Next type ***%WINDIR%\Installer**** and then select **Next**. There are no exceptions for this rule; therefore, you can click **Next** again.
26. Under name, make one up or just use "***Allow everyone to run all Windows Installer files located in the Windows\Installer folder***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
27. For the next rule, right-click **Windows Installer Rules** and select **Create New Rules**.
28. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Administrator***, press **ENTER**, and then select **Next**.
29. For Conditions, select **Path**, and then select **Next**.
30. Type "*****" then select **Next**. There are no exceptions for this rule; therefore, you can click **Next** again.
31. Under name, make one up or just use "***Allow administrators to run all Windows Installer files***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
32. For the next rule, right-click **Script Rules** and select **Create New Rules**.
33. Under **Permission**, select **Allow**. For **User or Group**, if ***Everyone*** is not the default, click **Select**, type ***Everyone***, press **ENTER**, and then select **Next**.
34. For Conditions, select **Path**, and then select **Next.**
35. Next type ***%PROGRAMFILES%**** and then select **Next**. There are no exceptions for this rule; therefore, you can click **Next** again.
36. Under name, make one up or just use "***Allow everyone to run all the scripts located in the***

Program File folder" and then click **Create**. If there's a permission window pop-up, then select **Yes**.

37. For the next rule, right-click **Script Rules** and select **Create New Rules**.
38. Under **Permission**, select **Allow**. For **User or Group**, if ***Everyone*** is not the default, click **Select** and type ***Everyone***, press **ENTER,** and then select **Next**.
39. For Conditions, select **Path**, and then select **Next**.
40. Next type ***%WINDIR%**** and then select **Next**.
41. For exceptions, set the **Add Exception** field to path.
42. Add the following exceptions by clicking the **Add** button.
    a. *%SYSTEM32%\catroot2**
    b. %SYSTEM32%\com\dmp*
    c. %SYSTEM32%\FxsTmp*
    d. %SYSTEM32%\spool\drivers\color*
    e. %SYSTEM32%\spool\PRINTERS*
    f. %SYSTEM32%\spool\SERVERS*
    g. %SYSTEM32%\Tasks*
    h. %WINDIR%\Debug*
    i. %WINDIR%\PCHEALTH\ERRORREP*
    j. %WINDIR%\Registration*
    k. %WINDIR%\SysWOW64\com\dmp*
    l. %WINDIR%\SysWOW64\FxsTmp*
    m. %WINDIR%\SysWOW64\Tasks*
    n. %WINDIR%\Tasks*
    o. %WINDIR%\Temp*
    p. %WINDIR%\tracing*
43. Under name, make one up or use "***Allow everyone to run all the scripts located in the Windows folder***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
44. For the next rule, right-click **Script Rules** and select **Create New Rules**.
45. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Administrator,*** press **ENTER** and then select **Next**.
46. For Conditions, select **Path**, and then select **Next**.
47. Next type "*****" and then select **Next**, there are no exceptions for this rule, so you can click **Next** again.
48. Under name, make one up or use "***Allow administrators to run all scripts***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
49. For the next rule, right-click **DLL Rules** and select **Create New Rules**.
50. Under **Permission**, select **Allow**. For **User or Group**, click **Select**, type ***Administrator***, press **ENTER**, and then select **Next**.
51. For Conditions, select **Path**, and then select **Next.**
52. Next type ***** and then select **Next**. There are no exceptions for this rule, so you can click **Next** again.
53. Under name, make one up or just use "***Allow administrators to execute all DLLs***" and then click

Create. If there's a permission window pop-up, then select **Yes**.

54. For the next rule, right-click **DLL Rules** and select **Create New Rules.**
55. Under **Permission**, select **Allow**. For **User or Group**, if ***Everyone*** is not the default, click **Select**, type ***Everyone***, press **ENTER**, and then select **Next**.
56. For Conditions, select **Path**, and then select **Next**.
57. Next type ***%PROGRAMFILES%**** and then select next. There are no exceptions for this rule so, you can click **Next** again.
58. Under name, make one up or just use "***Allow everyone to execute all DLLs located in the Program Files folder***" and then click **Create**. If there's a permission window pop-up, then select **Yes**.
59. For the next rule, right-click **DLL Rules** and select **Create New Rules**.
60. Under **Permission**, select **Allow**. For **User or Group**, if ***Everyone*** is not the default, click **Select**, type ***Everyone***, press **ENTER**, and then select **Next**.
61. For Conditions, select **Path**, and then select **Next**.
62. Next type ***%WINDIR%**** and then select **Next**"
63. For exceptions, set the **Add Exception** field to **Path**.
64. Add the following exceptions by clicking the **Add** button.
    a. %SYSTEM32%\catroot2*
    b. %SYSTEM32%\com\dmp*
    c. %SYSTEM32%\FxsTmp*
    d. %SYSTEM32%\spool\drivers\color*
    e. %SYSTEM32%\spool\PRINTERS*
    f. %SYSTEM32%\spool\SERVERS*
    g. %SYSTEM32%\Tasks*
    h. %WINDIR%\Debug*
    i. %WINDIR%\PCHEALTH\ERRORREP*
    j. %WINDIR%\Registration*
    k. %WINDIR%\SysWOW64\com\dmp*
    l. %WINDIR%\SysWOW64\FxsTmp*
    m. %WINDIR%\SysWOW64\Tasks*
    n. %WINDIR%\Tasks*
    o. %WINDIR%\Temp*
    p. %WINDIR%\tracing*
65. Under name, make up one or use the following: "***Allow everyone to execute all DLLs located in the Windows folder***", then click **Create**. If there's a permission window pop-up, then select **Yes**.

Appendix B – Enabling the DLL Rule Collection

Perform the following procedure to enable the DLL rule collection.

1. In the console tree for AppLocker, right-click **AppLocker**, and then click **Properties**.

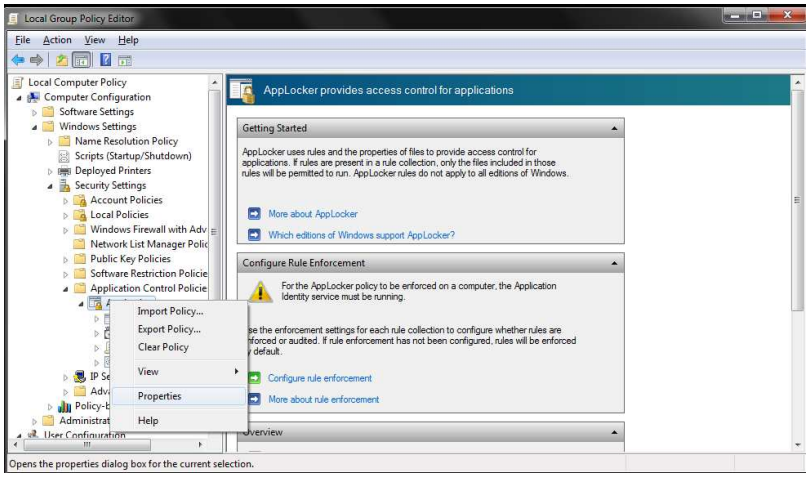


Figure 4: Application Control Policies > AppLocker

2. Click the **Advanced** tab, select the **Enable the DLL rule collection** check box and then click **OK**.

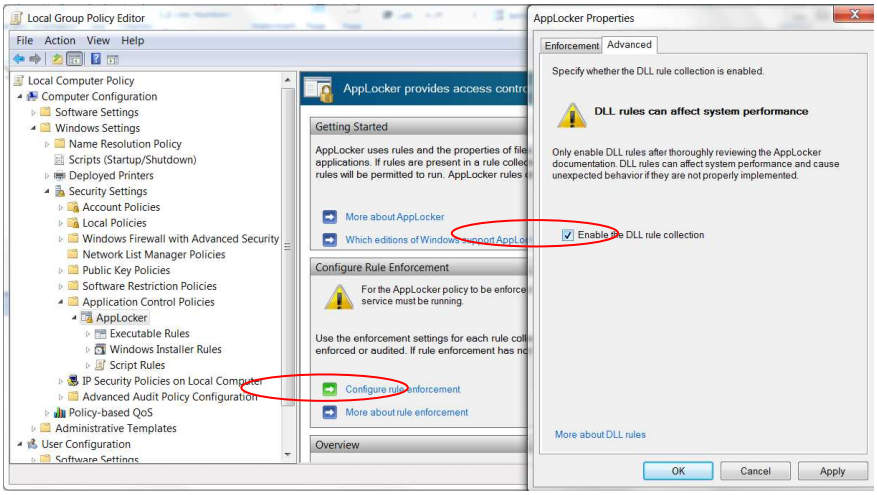


Figure 5: Enable DLL Rule Collection

Appendix C – AppLocker Rules to Prevent Administrators from Easily Browsing the Internet/Email

It is recommended that users with administrator privileges only log in as an administrator when administrator tasks are performed. The administrative accounts should not be used while browsing the web or reading emails. Therefore, a non-administrative account should be created for everyday tasks for internet and email usage. The settings in the table below are the recommended generic AppLocker rules for preventing administrators from easily browsing the Internet and reading email with administrative credentials. (Choose to deny access to whichever web browsers and email clients that are applicable to your organization.)

Location	Rule Name	Permissions Action	Permissions User or Group	Conditions Type	Conditions Value
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Internet Explorer web browser	Deny	BUILTIN\ Administrators	Publisher	O=MICROSOFT CORPORATION, L=REDMOND, S=WASHINGTON, C=US
additional publisher properties for the preceding rule				Product Name	WINDOWS® INTERNET EXPLORER
additional publisher properties for the preceding rule				File Name	IEXPLORE.EXE
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Outlook email client	Deny	BUILTIN\ Administrators	Publisher	O=MICROSOFT CORPORATION, L=REDMOND, S=WASHINGTON, C=US
additional publisher properties for the preceding rule				Product Name	MICROSOFT OFFICE OUTLOOK
additional publisher properties for the preceding rule				File Name	OUTLOOK.EXE
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Thunderbird email client	Deny	BUILTIN\ Administrators	Publisher	O=MOZILLA MESSAGING INC., L=MOUNTAIN VIEW, S=CALIFORNIA, C=US
additional publisher properties for the preceding rule				Product Name	THUNDERBIRD
additional publisher properties for the preceding rule				File Name	thunderbird.exe
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Chrome web browser	Deny	BUILTIN\ Administrators	Publisher	O=GOOGLE INC, L=MOUNTAIN VIEW, S=CALIFORNIA, C=US
additional publisher properties for the preceding rule				Product Name	GOOGLE CHROME
additional publisher properties for the preceding rule				File Name	chrome.exe

Location	Rule Name	Permissions Action	Permissions User or Group	Conditions Type	Conditions Value
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Firefox web browser	Deny	BUILTIN\ Administrators	Publisher	O=MOZILLA CORPORATION, L=MOUNTAIN VIEW, S=CALIFORNIA, C=US
additional publisher properties for the preceding rule				Product Name	FIREFOX
additional publisher properties for the preceding rule				File Name	firefox.exe
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Opera web browser	Deny	BUILTIN\ Administrators	Publisher	O=OPERA SOFTWARE ASA, S=OSLO, C=NO
additional publisher properties for the preceding rule				Product Name	OPERA
additional publisher properties for the preceding rule				File Name	opera.exe
Security Settings\Application Control Policies\AppLocker\ Executable Rules	Prevent administrators from easily running the Safari web browser	Deny	BUILTIN\ Administrators	Publisher	O=APPLE INC., L=CUPERTINO, S=CALIFORNIA, C=US
additional publisher properties for the preceding rule				Product Name	SAFARI
additional publisher properties for the preceding rule				File Name	Safari.exe

Table 8: Generic Admin AppLocker Rules

1. If you are already in AppLocker under **Group Policy Editor** skip to step 3 or else click **Start**, type ***gpedit.msc*** and then press **ENTER**.
2. Next, in the console, under the **Local Computer > Windows Settings > Security Settings > Application Controls > AppLocker**, then highlight **Executable Rules** and then right-click **Executable Rules**.
3. Select **Create New Rules** and under Permission, select **Deny.** For User or Group click **Select**, type ***Administrator***, press **ENTER,** and then select **Next**.
4. For Conditions, select **Publisher**, and then select **Next**.
5. In browse, search for Internet Explorer's "IEXPLORE.EXE" main executable file. Once the file is found, select it and the remainder of the information should automatically fill in.

Appendix D – Creating and Modifying AppLocker Rules

There are several different ways to create or modify AppLocker Rules. You can use either AppLocker Local Group Policy, Group Policy Management or Local Security Policy Editor. Use whichever editor that best meets your organization or individual needs.

Create AppLocker Rules

1. Within AppLocker, right-click the rule collection of the preferable rule (Executable, Windows Installer, or Script) to create the desired rule and then click **Create New Rule**.

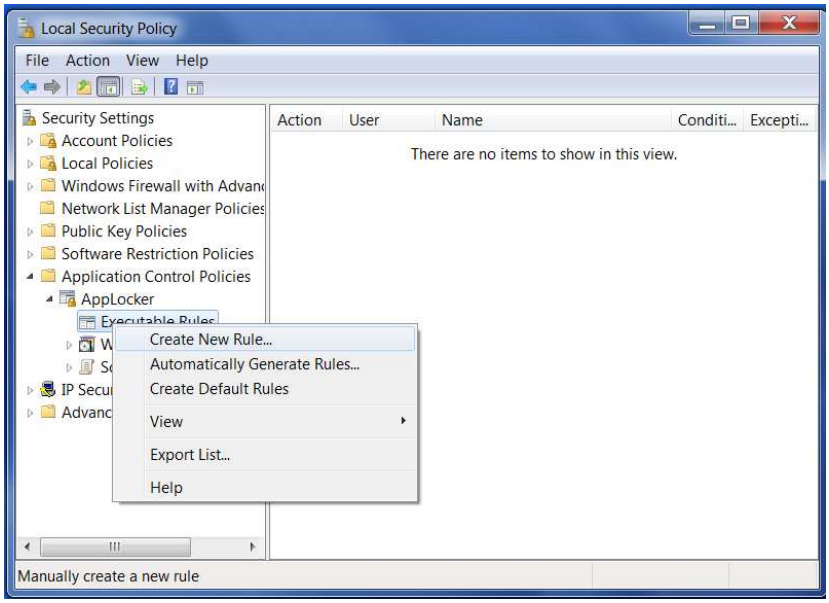


Figure 6: Create New Rule

2. Click **Allow** or **Deny** to allow or deny the files contained within the rule.
3. Click **Select** for the **Select User or Group** box, then type the appropriate group or user and then click **OK**.
4. Click **Next**.
5. Click the appropriate rule condition for this rule. You can choose from **Publisher**, **Path**, or **File hash** and then click **Next**.

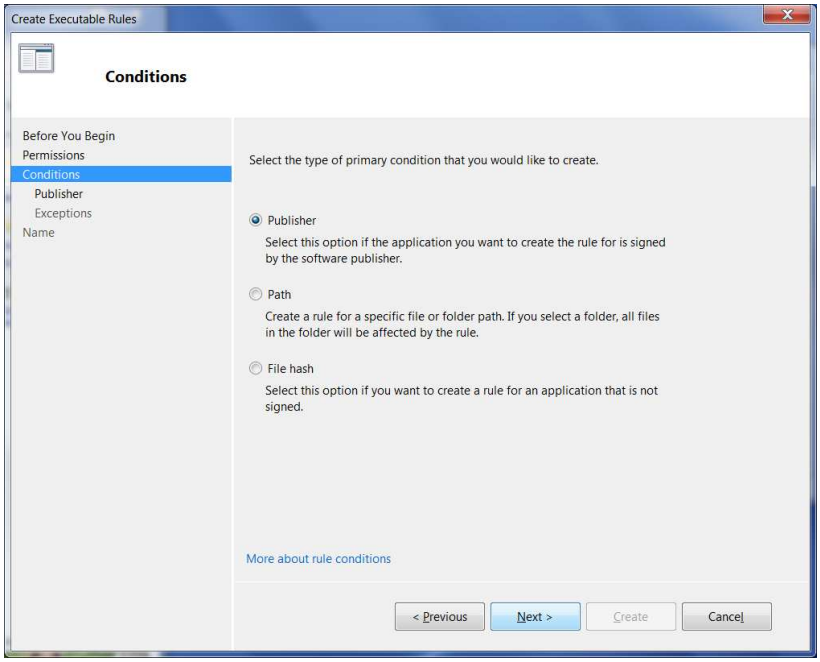


Figure 7: Conditions

6. Depending on the rule condition selected, there are different criteria:

    - **Publisher rule condition**. Click **Browse** to select the file to extract the publisher information. To edit the publisher information, select the **Use custom values** check box and then edit the values. Click **Next**.

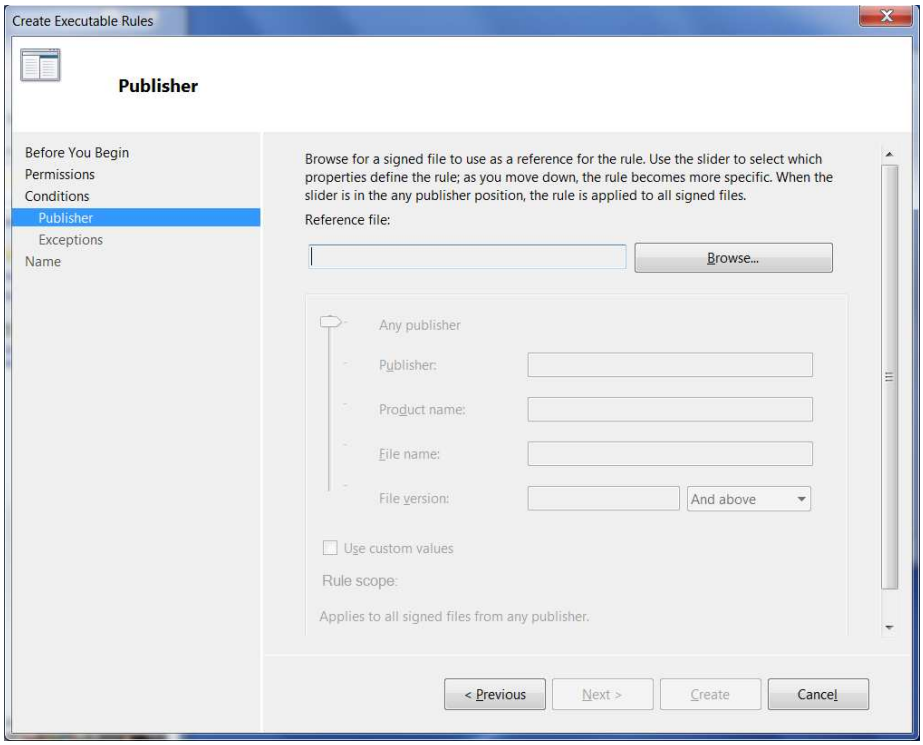


Figure 8: Publisher

- **Path rule condition**. Navigate to the file or folder by clicking **Browse Folders** or **Browse Files**. Optionally, type the path into the **Path** box. Click **Next**.

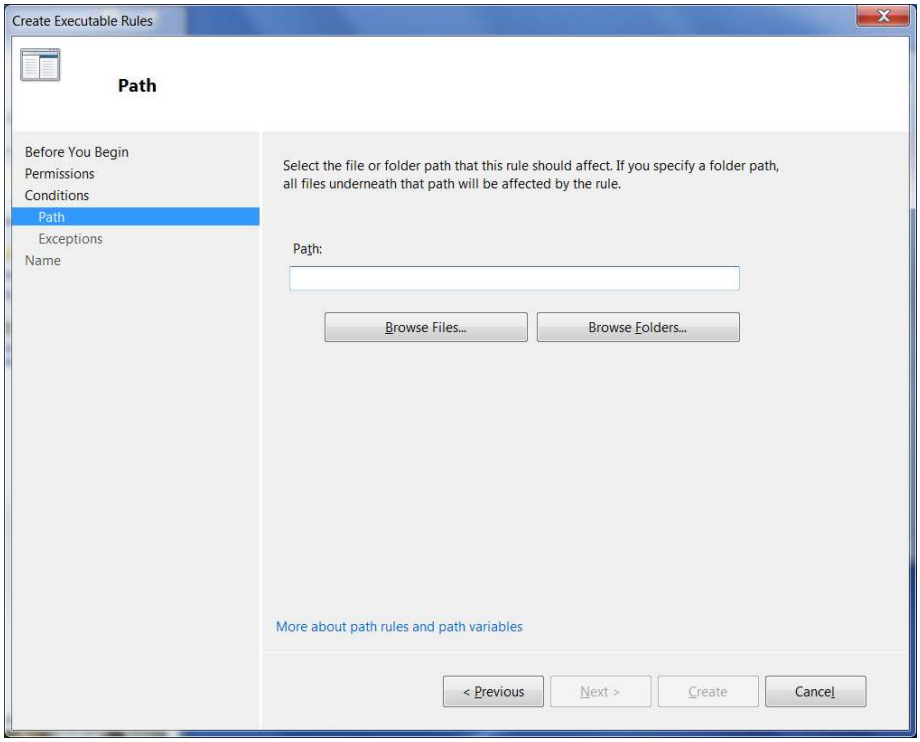


Figure 9: Path

- **File hash rule condition**. Navigate to the file or folder by clicking **Browse Folders** or **Browse Files**. Click **Next**.

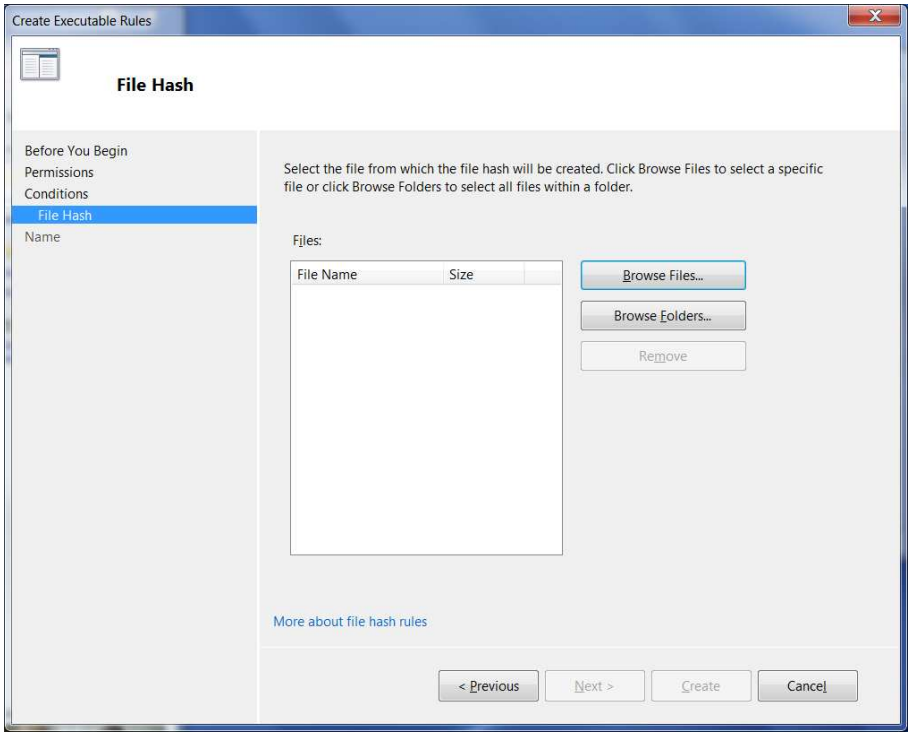


Figure 10: File Hash

Optional Steps:

1. On the **Exceptions** page, specify the publisher or path to exclude from the rule, and then click **Next**. (Note: You cannot create exceptions for the hash rules.)
2. In the **Name** box, type a name to identify the rule.
3. In the **Description** box, type a description that explains the purpose of this rule.
4. Click **Create**.

Edit AppLocker Rules

The **Properties** dialog box displays several different tabs based on the primary condition of the rule. A rule with a publisher primary condition includes the **General**, **Publisher**, and **Exceptions** tabs. The **Exceptions** tab is not displayed for rules with a primary condition set to file hash. A rule with a path primary condition includes the **General**, **Path**, and **Exceptions** tabs. A rule with a file hash primary condition includes the **General** and **File Hash** tabs.

It's recommended not to edit AppLocker rule collection while it's being enforced in Group Policy since making changes to live policies may result in unexpected behaviors. The following explains the details on each tab:

General tab

On the **General** tab, the name and description of the rule can change, the action of the rule of allow or deny can change and the user or group to which the rule applies can change.

Publisher tab

The **Publisher** tab is available only for rules that have publisher conditions. On the **Publisher** tab, the publisher name, product name, file name and file version to which the rule applies can change.

Path tab

The **Path** tab is available only for rules that have path conditions. On the **Path** tab, the folder or file path to which the rule applies can change.

File Hash tab

File hash is available to create a rule that applies to a specific file. The **File Hash** tab is for rules that have file hash conditions. On the **File Hash** tab, one can add and remove files that are included in the rule.

Exceptions tab

On the **Exceptions** tab, one can add or edit files or folders that are excluded from the rule. As mentioned before, the **Exceptions** tab is not available for file hash rules.

Editing Publisher Rule

1. Within the AppLocker Group Policy Editor, click the appropriate rule collection.

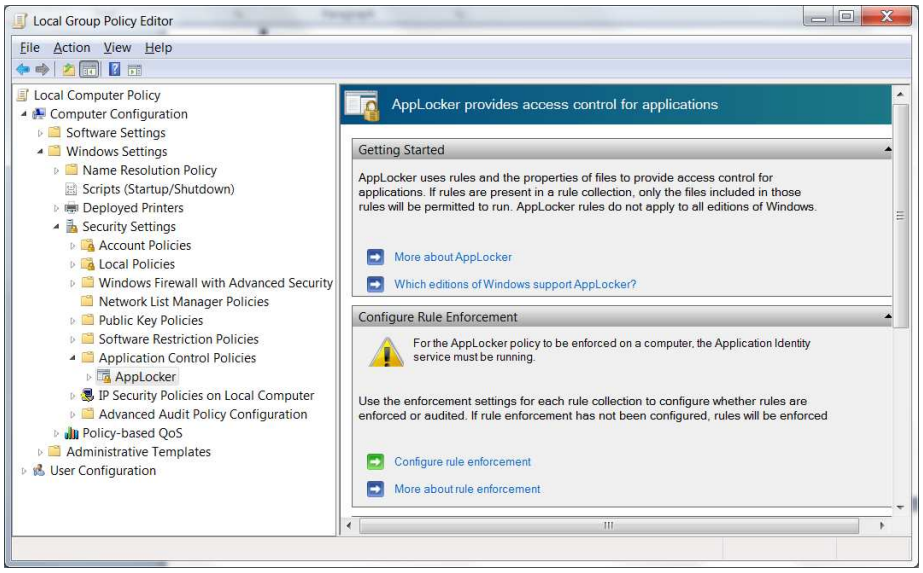


Figure 11: Local Group Policy Editor (AppLocker)

2. In the **Action** pane, right-click the publisher's rule, and then click **Properties**.

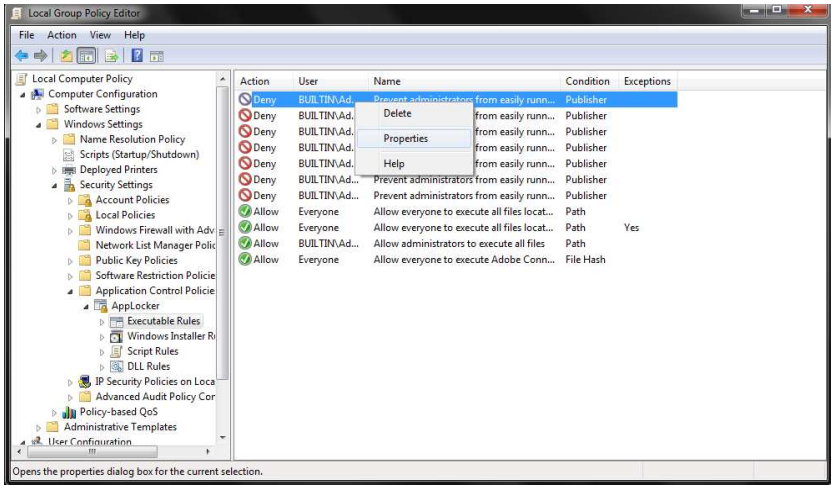


Figure 12: Right-click Publisher Rule

3. Click the appropriate tab to edit the rule properties.
    - Click the **General** tab to change the rule name, add a rule description, configure whether the rule is used to allow or deny applications, and set the security group for which this rule should apply.

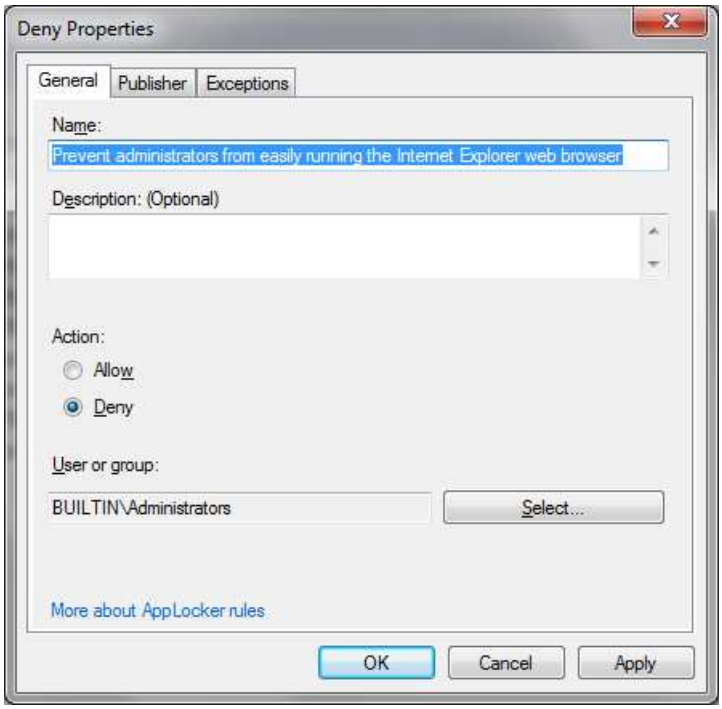


Figure 13: Rule Properties

    - Click the **Publisher** tab to configure the certificate's common name, the product name, the file name, or file version (asterisk, *, indicates all file version) of the publisher.

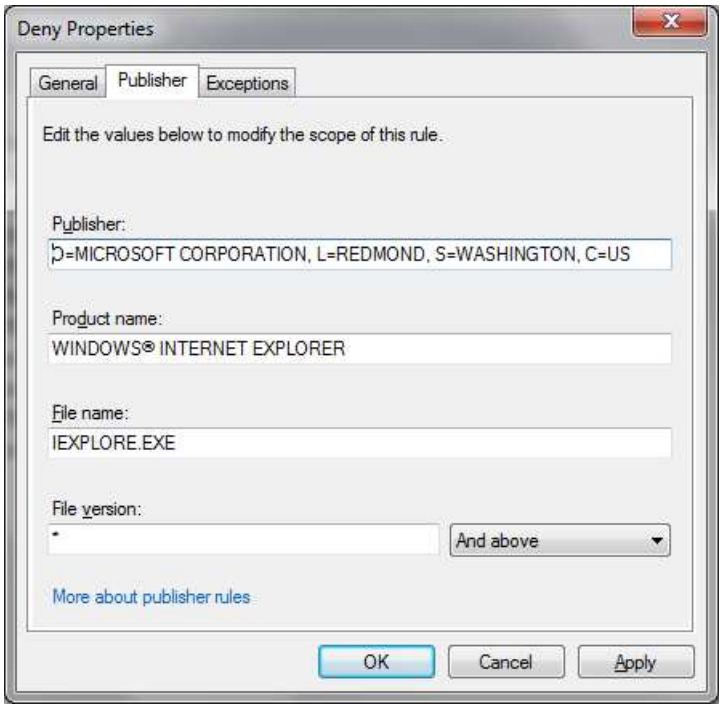


Figure 14: Publisher Tab

- Click the **Exceptions** tab to create or edit exceptions.

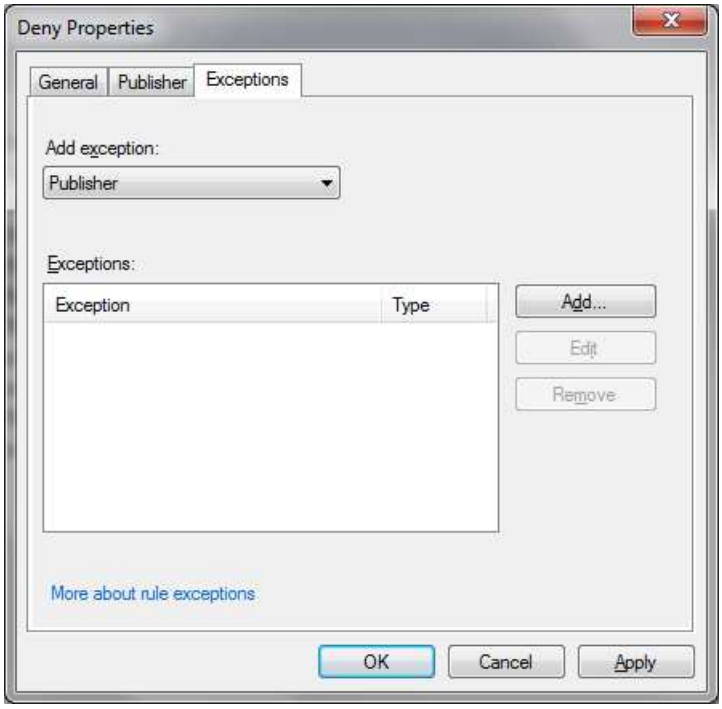


Figure 15: Exceptions Tab

- When you finish updating the rule, click **OK**.

Editing File Hash Rule

1. Within the AppLocker Group Policy Editor, click the appropriate rule collection.

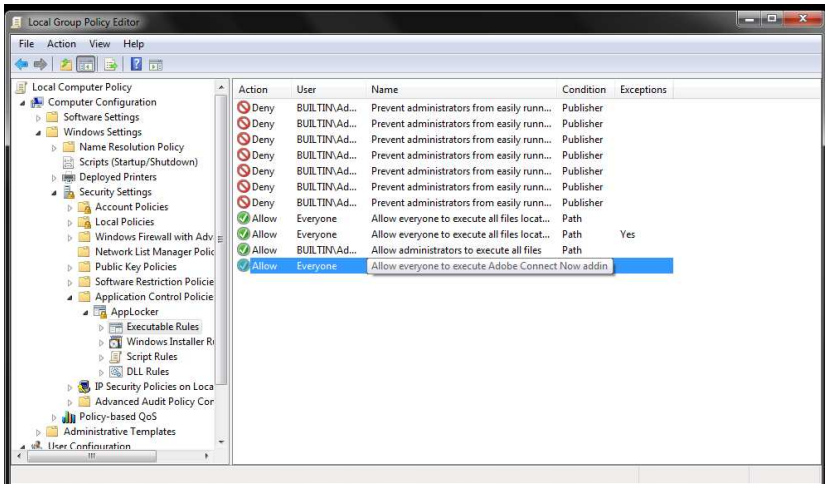


Figure 16: Select File Hash Rule

2. In the **Action** pane, right-click the file hash rule, and then click **Properties**.
3. Click the appropriate tab to edit the rule properties.
    - Click the **General** tab to change the rule name, add a rule description, configure whether the rule is used to allow or deny applications, and set the security group in which this rule should apply.

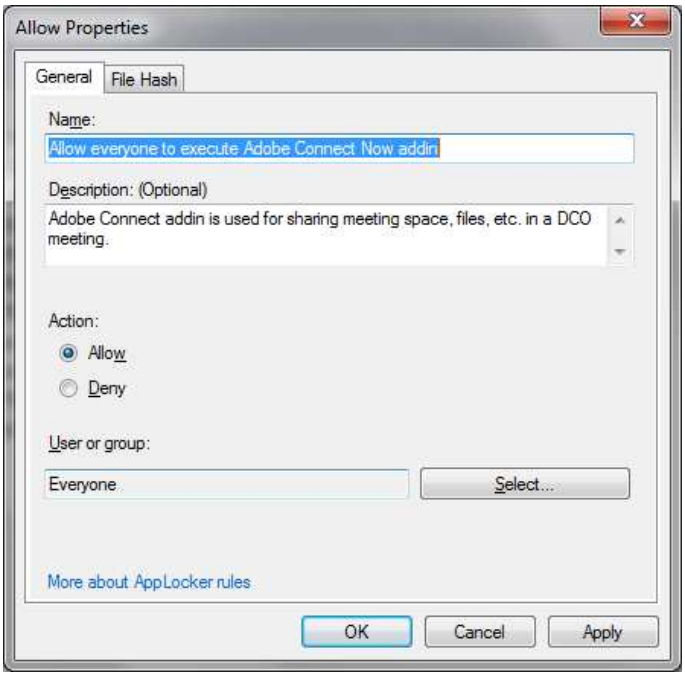


Figure 17: General Tab in File Hash Properties

- Click the **File Hash** tab to configure the files that should be used to enforce the rule. You can use the **Browse Files** button to add a specific file or the **Browse Folders** button to add all files in a specified folder.

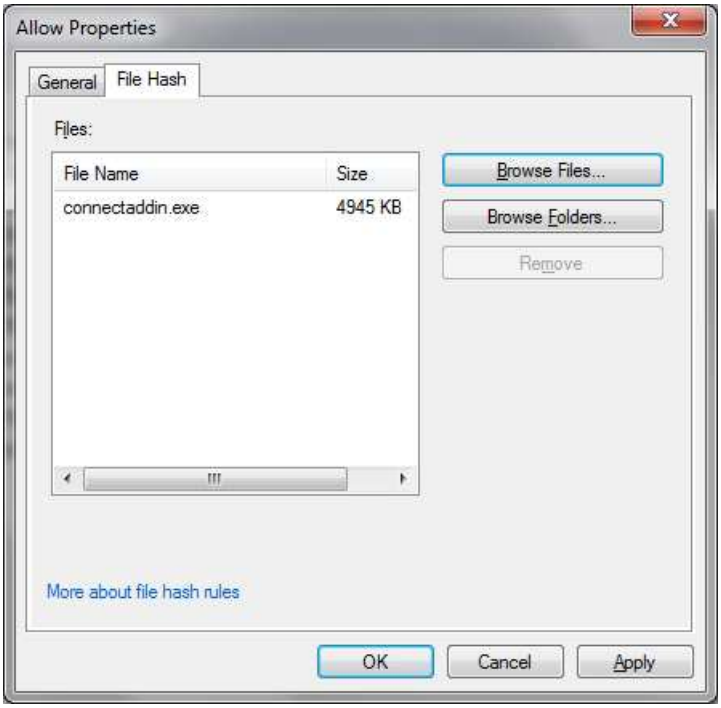


Figure 18: File Hash Tab in File Hash Properties

- When you finish updating the rule, click **OK**.

Editing Path Rule

1. Within the AppLocker Group Policy Editor, click the appropriate rule collection.

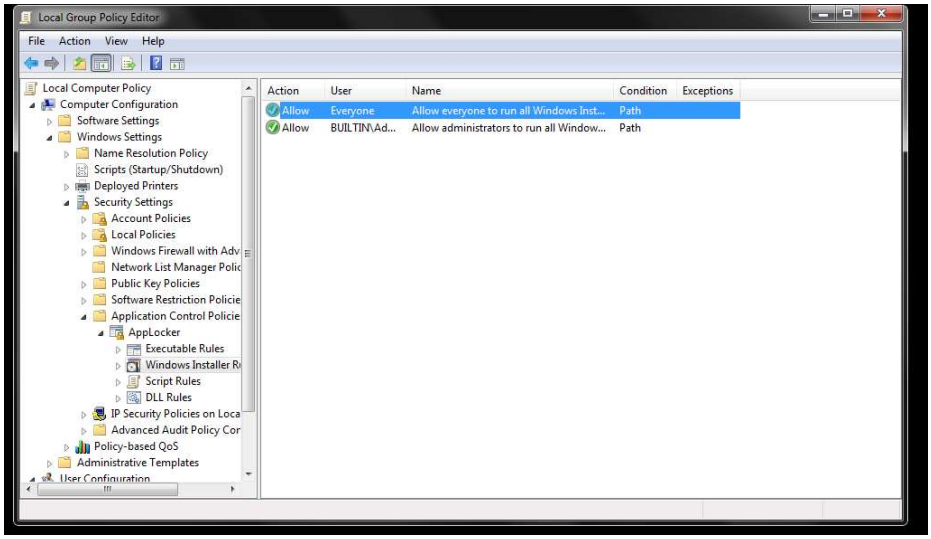


Figure 19: Edit Path Rule

2. In the **Action** pane, right-click the path rule and then click **Properties**.

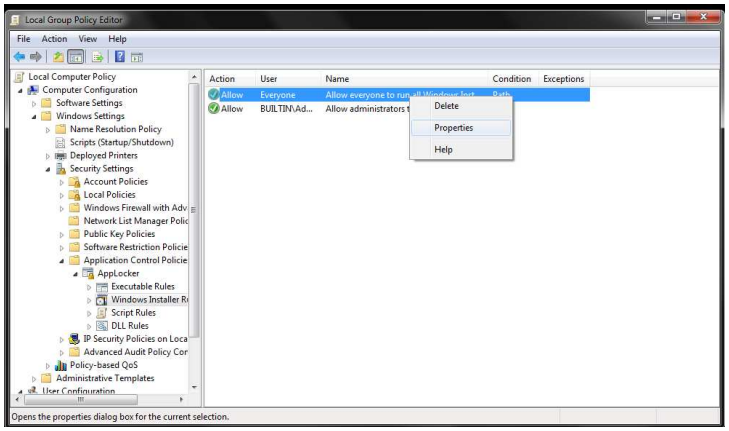


Figure 20: Select Properties for Path Rule

3. Click the appropriate tab to edit the rule properties.
   - Click the **General** tab to change the rule name, add a rule description, configure whether the rule is used to allow or deny applications, and set the security group in which this rule should apply.

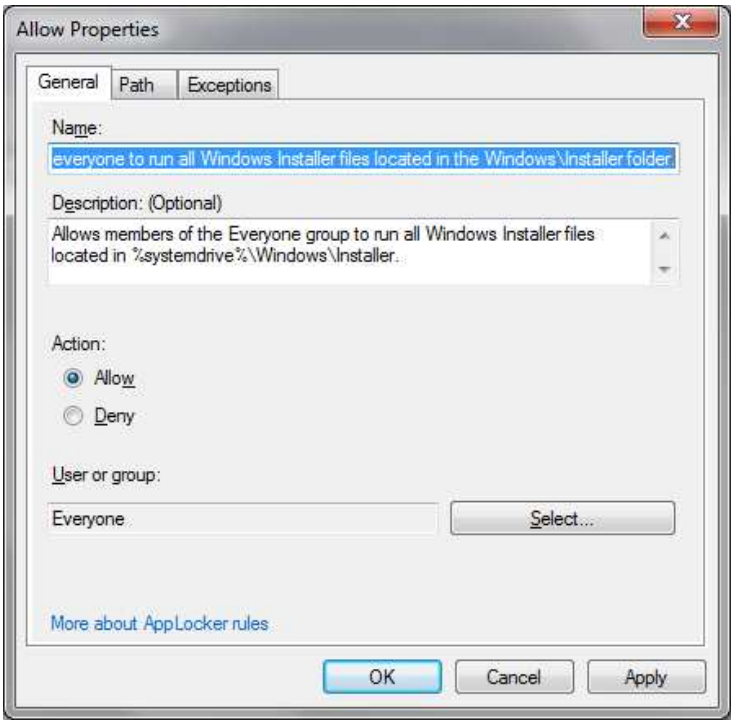


Figure 21: General Tab for Path Rule Properties

   - Click the **Path** tab to configure the path on the computer in which the rule should be enforced.

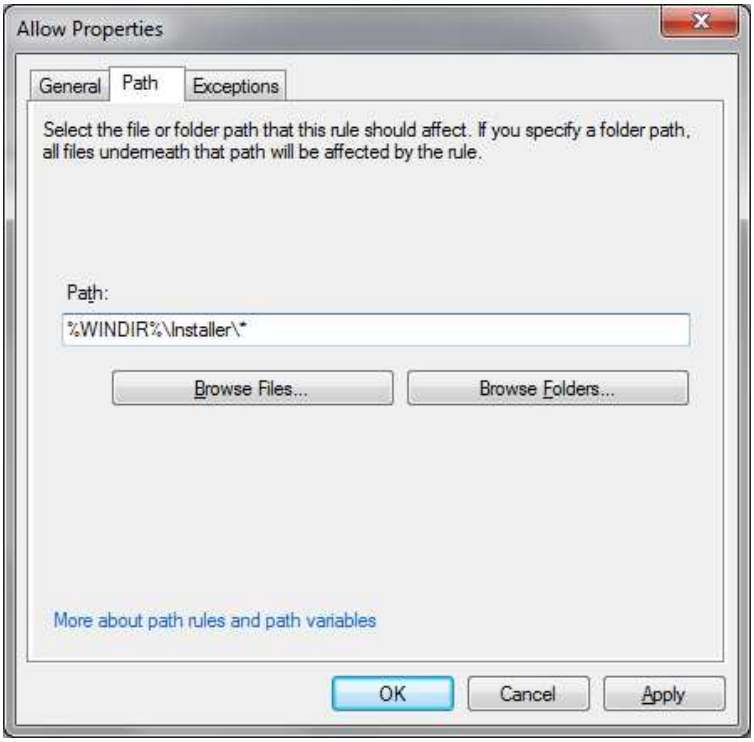


Figure 22: Path Tab for Path Rule Properties

- Click the **Exceptions** tab to create exceptions for specific files in a folder.

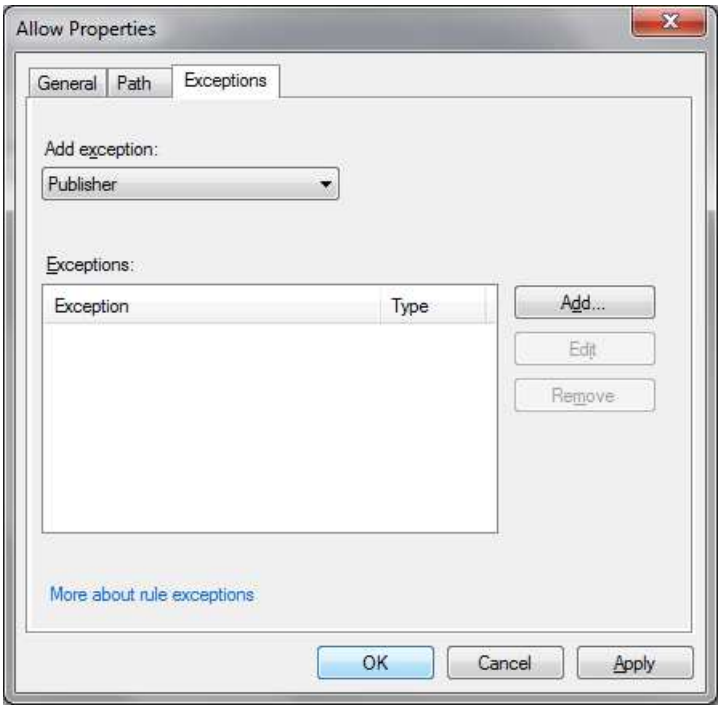


Figure 23: Exceptions Tab for Path Rule Properties

- After completing the update for the rule, click **OK**.

Delete AppLocker Rules

1. Within the AppLocker Group Policy Editor, click the appropriate rule collection.

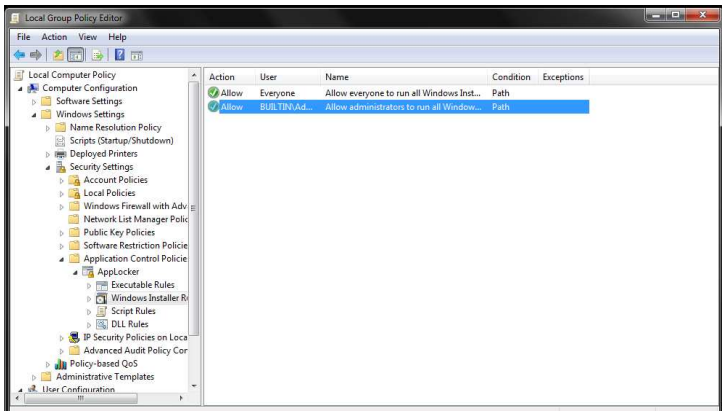


Figure 24: Delete an AppLocker Rule

2. In the details pane, right-click on the rule, click **Delete** and then click **Yes**.

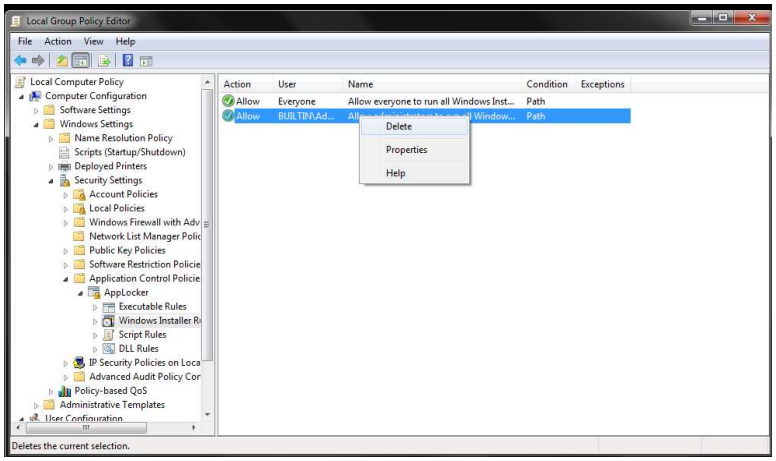


Figure 25: Select Delete

Appendix E – Using a Task Script to Gather Process and User Information

When using DLL rules with event forwarding, it is beneficial for the AppLocker events to actually list the path of the process executable, instead of just the process ID of the process that triggered the event. This information can also be very useful for troubleshooting when an application is not functioning correctly and it is not clear whether AppLocker is the cause. A script is provided with this guide that can be triggered based on an AppLocker event to search through Process Creation events to find the corresponding process that triggered the AppLocker event. Then include that information in a new meta-event. These new meta-events can also be forwarded on to a central server; several scripts are also included to aid analysis of these events.

Appendix F – Helper Files Included with This Guide

The following is a list and brief description of each helper file that is included with this guide. All of the script files (.bat, .ps1) have the .txt extension added for inclusion with this guide so they will not be accidentally executed. ***The .txt extension should be removed before using the scripts. The scripts associated with this guide can be found on the IAD GitHub site at https://github.com/iadgov***

AppLocker Starter Policy

1. *AppLocker Starter Policy.xml*
   This file is an AppLocker Policy xml file that can be imported using the Group Policy Management Editor to automatically configure a starting location-based application whitelisting policy for initial auditing. Once applied, the events will need to be reviewed to tailor the policy to the network.

Create AppLocker Popup Task

2. *Create AppLocker Popup Task.bat.txt*
   This script can be pushed out as a computer startup script to create the "Create AppLocker Popup" that informs the user that an AppLocker event has triggered and that an administrator should be notified. This file should be placed in the \\<domain>\SYSVOL\<domain>\Scripts file share along with the following task xml file in order to create the task successfully.
3. *AppLocker Popup Alert Task.xml*
   This file is a Task Scheduler task xml file that sets itself to be triggered by an AppLocker warning or error event and then displays a popup to the user, notifying them that AppLocker has been triggered on a file and to notify their administrator.

Event Viewer AppLocker Custom View

4. *AppLocker Event Viewer Custom View.xml*
   This file is an Event Viewer custom view that can be imported to automatically filter and show only the AppLocker warning and error events.

AppLocker Event Forwarding

5. *AppLocker Event Forwarding Subscription Query Filter xml.txt*
   When setting up an event forwarding subscription, it is necessary to tell the subscription which events are desired for forwarding. This file contains an event query that can be used within an event subscription to select only the AppLocker warning and error events for forwarding.
6. *AppLocker Events to CSV.ps1.txt*
   This script can be run on a server that collects forwarded AppLocker events to help perform basic analysis by converting the events into a CSV (comma separated value) format that can be imported in Excel for analysis.
7. *AppLocker Events Grouped by File.ps1.txt*
   This script can be run on a server that collects forwarded AppLocker events to help perform basic analysis by grouping the events by file path to show if there are any files that are being blocked many times.

Create AppLocker Meta Events

8. *Create AppLocker Meta Event Task.bat.txt*
   This script can be pushed out as a computer startup script to create the "Create AppLocker Meta Event" task on each computer to trigger when an AppLocker event is created and run a script to create a new AppLocker meta event with additional calling process and user information. This file should be placed in the \\<domain>\SYSVOL\<domain>\Scripts file share along with the following PowerShell script and task xml files in order to create the task successfully.
9. *Create AppLocker Meta Event.ps1.txt*
   This script should be triggered by an AppLocker event and given the corresponding event's EventRecordID as an argument on the commend line. It then finds that event in the normal AppLocker log, retrieves the ProcessID listed in the event, searches through the Security event log for process creation events that match that ProcessID, and then creates a new meta event in a new AppLocker log that contains the original event data, as well as the path of the calling process and the username of the user running the process that triggered the event. This additional information should aid troubleshooting and investigations based on AppLocker events.
10. *Create AppLocker Meta Event Task.xml*
    This file is a Task Scheduler task xml file that sets itself to be triggered by an AppLocker warning or error event and then executes the "C:\Windows\Scripts\Create AppLocker Meta Event.ps1" PowerShell script to create a new AppLocker meta event with the calling process' path information and the user's username.
11. *AppLocker Meta Events Custom View.xml*
    This file is an Event Viewer custom view that can be imported to automatically filter and show only the AppLocker Meta Events.
12. *AppLocker Event Forwarding Subscription Meta Event Query Filter xml.txt*
    When setting up an event forwarding subscription, it is necessary to tell the subscription which events are desired for forwarding. This file contains an event query that can be used within an event subscription to select the AppLocker Meta Events for forwarding.
13. *AppLocker Meta Events to CSV.ps1.txt*
    This script can be run on a server that collects forwarded AppLocker Meta events to help perform basic analysis by converting the events into a CSV (comma separated value) format that can be imported in Excel for analysis.
14. *AppLocker Meta Events Grouped by File.ps1.txt*
    This script can be run on a server that collects forwarded AppLocker Meta events to help perform basic analysis by grouping the events by file path to show if there are any files that are being blocked many times.

Appendix G – Packaged App Rule

The following instructions explain how to create a whitelist to allow only certain Packaged Apps to run:

1. Within AppLocker, right-click the Packaged app Rules settings and then click **Create New Rule**.
2. Click **Allow** or **Deny** to allow or deny the packaged apps contained within the rule.
3. Click **Select** in the **Select User or Group** box, type the appropriate group or user and the click **OK**.
4. Click **Next**.
5. Select either **Use an installed package app as a reference** or **Use a package app installer as a reference.** Then select either the packaged app or the .appx file associated with a packaged app. Verify that the **Publisher, Package Name, and Package Version** are correct. These can be manually edited by checking the **Use custom values** check box.
6. Click **Next.**
7. If exceptions are needed to the packaged app rule[6], press the **Add** button then follow the wizard to select an application using either **Use an installed package app as a reference** or **Use a package app installer as a reference. S**elect either the packaged app or the .appx file associated with a packaged app. Verify that the **Publisher, Package Name, and Package Version** are correct. These can be manually edited by checking the **Use custom values** check box.
8. Click **Next.**
9. Assign a **Name** and **Description** (Optional) to the new rule, then click **Create.**

[6] Exceptions allow you to exclude packaged apps that would be normally included in the rule.

If you do not know the Package apps on your system, use the "Automatically Generate Rules…" option.

1. Within the AppLocker Group Policy Editor, right click the **Packaged app Rules**, and then click **Automatically Generate Rules…** option. After completing **Application and Permissions** with the desired settings click **Next.**
2. Confirm the **Reduce the number of rules…**
3. Review and select your **allowed apps rules** and then click **Create.**

Note:

- To restrict users from downloading new apps from the app store, deny access to the Metro Control Panel (*windows.immersivecontrolpanel*) and the Windows App Store (*WinStore*) apps.
- If using the automatically generated rules, some Packaged apps may not be listed since some application packages are framework packages that other apps leverage. Blocking these packages may inadvertently cause app failures for desired apps, therefore, AppLocker automatically filters packages that are registered as framework packages.
- In cases where AppLocker policies were created in older Windows versions (without the Packaged apps Rules) and an Exe rule collection exists in the domain policy, AppLocker by default blocks all Packaged apps on new computers joining the domain running Windows 8 and above. In these cases, if you want Packaged apps to run in your enterprise, it must be configured to allow it.